MATLAB- wprowadzenie

MATLAB - jest środowiskiem obliczeniowym przeznaczonym dla inżynierów i naukowców, umożliwiającym przeprowadzanie obliczeń matematycznych, analizy numerycznej, wizualizacji otrzymanych wyników (2D, 3D), jak również tworzenie algorytmów i programów. Język MATLABa jest intuicyjny i wygodny w użyciu, co sprawia, że opracowanie algorytmów jest prostsze niż w przypadku takich języków programowania jak C czy Fortran. Matlab może być instalowany na komputerach pracowniczych jak i w laboratoriach komputerowych.


AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
AGH
Centrum Rozwiązań Informatycznych AGH
System SkOs
Porady CRI
Pomoc IT
Regulamin USK
Historia AGH
CRI » Usługi » Oprogramowanie » Licencje dostępne dla pracowników i studentów
› CRI
› Aktualności
› POWER 3.5
› Licencje i oprogramowanie
› Usługi
› Poczta elektroniczna
› Strony WWW
› Sieć bezprzewodowa
› Sieć przewodowa
› Chmura AGH
› VPN
› Bazy danych
› Oprogramowanie
Licencje dostępne zarówno dla pracowników jak i studentów z możliwością instalacji na prywatnych komputerach:
MATLAB - jest środowiskiem obliczeniowym przeznaczonym dla inżynierów i naukowców, umożliwiającym przeprowadzanie obliczeń matematycznych, analizy numerycznej, wizualizacji otrzymanych wyników (2D, 3D), jak również tworzenie algorytmów i programów. Język MATLABa jest intuicyjny i wygodny w użyciu, co sprawia, że opracowanie algorytmów jest prostsze niż w przypadku takich języków programowania jak C czy Fortran. Matlab może być instalowany na komputerach pracowniczych jak i w laboratoriach komputerowych.
Informacje na stronie http://home.agh.edu.pl/~matlab.

http://home.agh.edu.pl/~matlab/Matlab_AGH/Informacje_podstawowe.html

INFORMACJE PODSTAWOWE ZGŁOSZENIE I REJESTRACJA DOSTĘP DO NOŚNIKA INSTALACJA FAQ

INFORMACJE PODSTAWOWE

Akademia Górniczo-Hutnicza zakupiła obecnie licencje w ramach:
MathWorks Total Academic Headcount
Model ten zapewnia możliwość korzystania bez limitu z wszystkich dostępnych toolboxów.
W ramach tego podejścia w szczególności dostępne są licencje:
- licencja individual,
- licencja classroom.

UWAGA:
Obecne LICENCJE pozwalają na pracę NIE TYLKO pracownikom AGH ale i studentom.
(Szczegóły w sekcji FAQ)

Licencja Individual

Powinna być wykorzystywana tylko i wyłącznie na potrzeby pracy naukowo-badawczej oraz nauki przez studentów i nie może być wykorzystywana do prac komercyjnych.
Jedna osoba może zainstalować aplikację na maksymalnie 4ch swoich urządzeniach.

Licencja Classroom

Celem licencji jest prowadzenie zajęć dydaktycznych w laboratoriach.

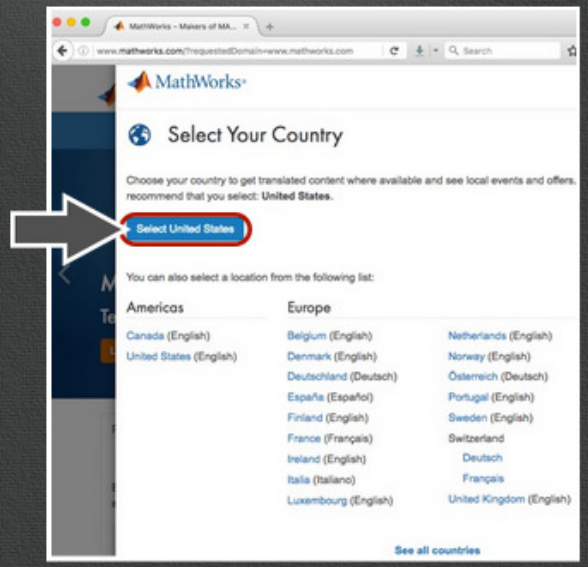
INFORMACJE PODSTAWOWE ZGŁOSZENIE I REJESTRACJA DOSTĘP DO NOŚNIKA INSTALACJA FAQ

INSTALACJA

I. UZYSKANIE DOSTĘPU DO NOŚNIKA

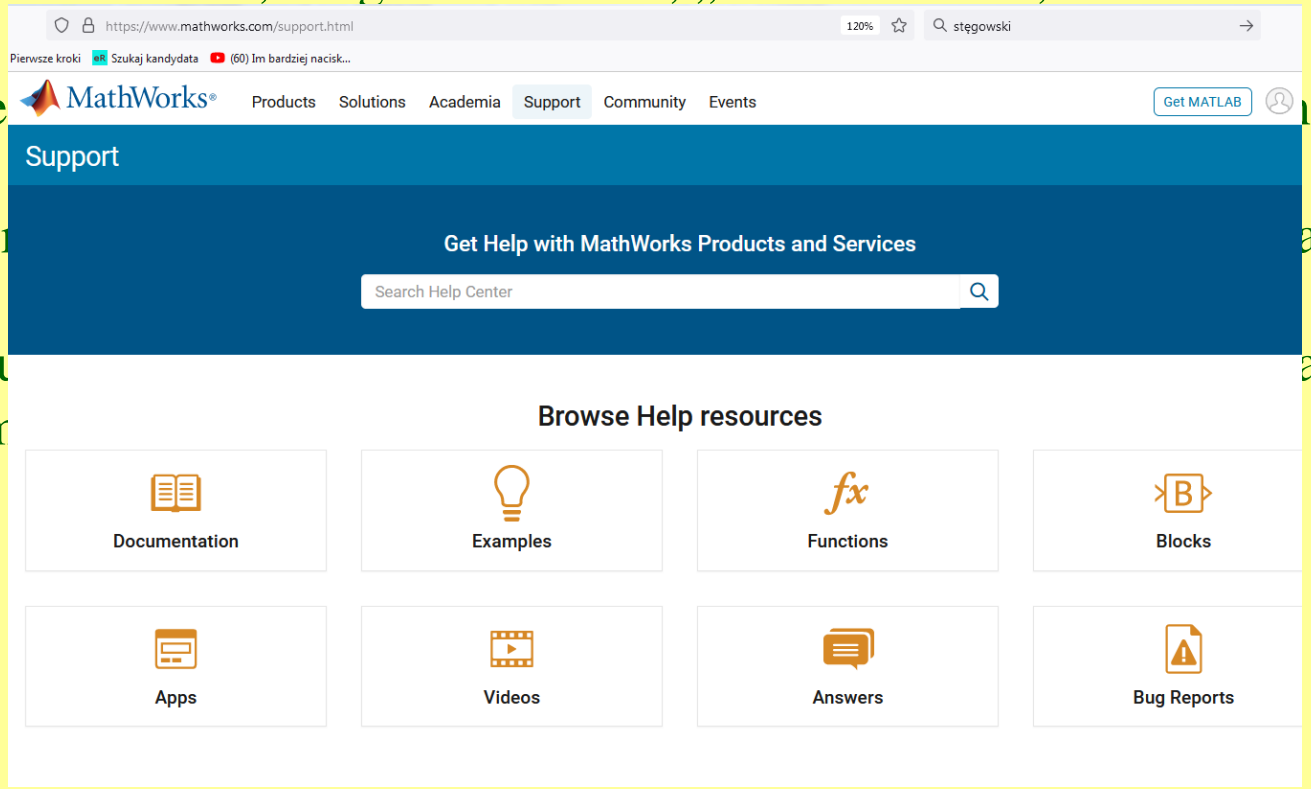
Poniższe podpunkty pokazują jak:
- założyć konto na serwerze MathWorks
- przypisać licencję
- zapoznać się z dostępnymi toolboxami w ramach licencji
- pobrać instalator

1. Należy wejść na stronę: http://www.mathworks.com
Jeśli pojawi się ekran jak poniżej należy zaznaczyć: Select United States
Ewentualnie od razu może pojawić się ekran jak w punkcie 2.

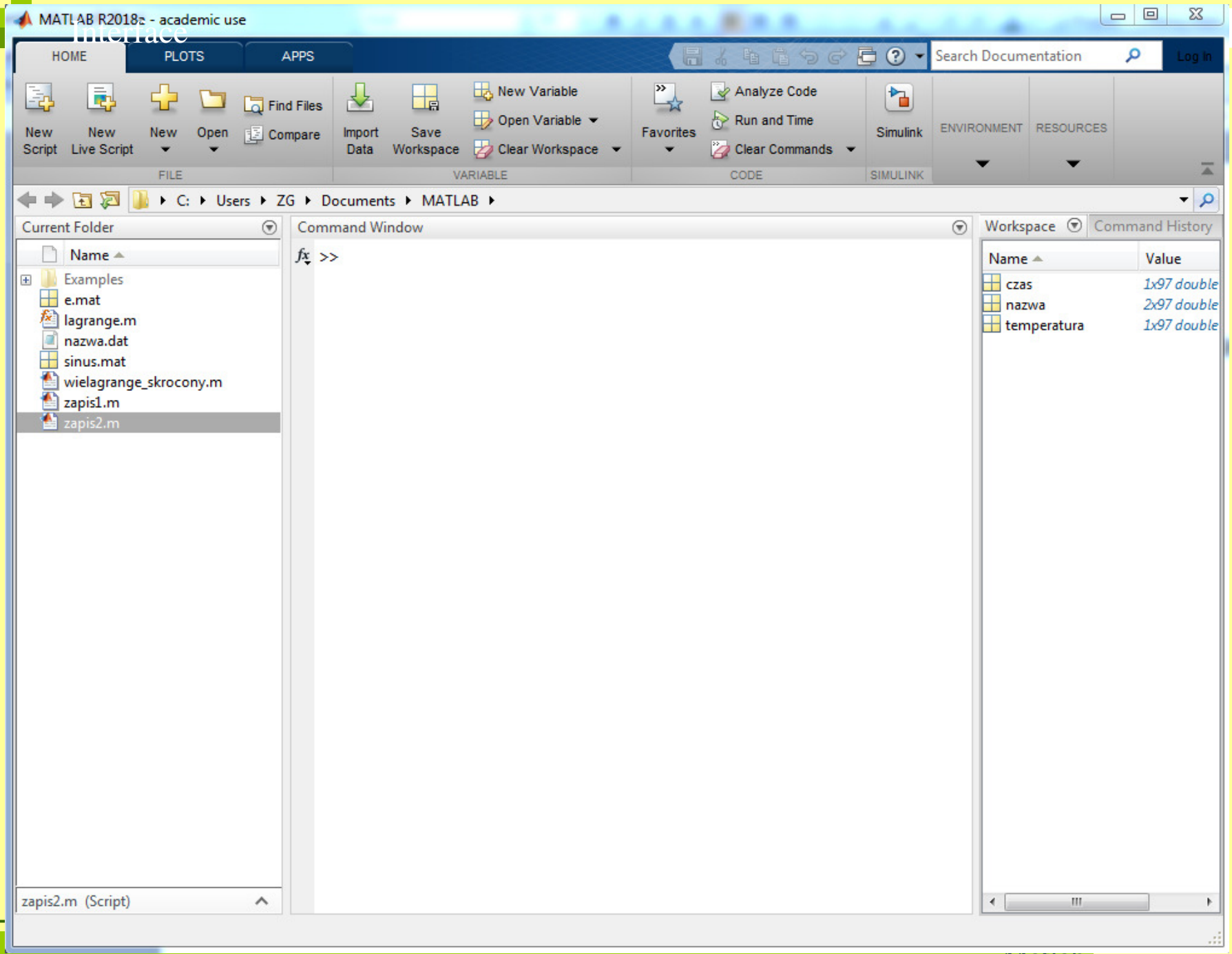


Literatura:

1. Jerzy Brzózka, Lech Dorobczyński, „Programowanie w MATLAB”,wyd. Mikom, 1998.
2. Marek Czajka, „Ćwiczenia Matlab”, wyd. Helion, 2005.
3. Bogumiła Mrozek, Zbigniew Mrozek, „MATLAB 5.X, SIMULINK 2.X”, PLJ, 1998
4. Walde ...ia”, Helion, 2015.
5. Wikto ...alma Press. 2009
6. Zygmu ...azów z zadaniam



Interface



Matlab

System pomocy

okienko ***Search Documentation***
(przykłady, live script)

ikona ***Help → Documentation*** ale też ***Help → Support Web Site***

W linii komend piszemy:
>>help funkcja
Jeśli nie znamy całej nazwy piszemy:

>>help fun… + tabulator

Podstawowe komendy

who - informacja o dostępnych zmiennych, same nazwy
whos - pełna informacja o dostępnych zmiennych
clc - czyszczenie okna komend
clear a - usuniecie z przestrzeni roboczej zmiennej a
clear all - usuniecie wszystkich zmiennych
ver - podaje numer wersji Matlaba oraz numery zainstalowanych dodatków
demo - wyświetla dostępne przykłady
bench - sprawdzenie szybkości pracy Matlaba - benchmark
help funkcja - wypisuje pomoc dotyczącą funkcji funkcja

Symbole operatorów

= Przypisanie wartości
[] Tworzenie macierzy, list argumentów wyjściowych funkcji
() Listy argumentów wejściowych funkcji, kolejność działań matematycznych
. Kropka dziesiętna, część operatorów arytmetycznych
.. Katalog macierzysty
... Kontynuacja polecenia jest w następnej linii
, . Symbole separacji argumentów funkcji, indeksów, itp.
; Koniec wiersza macierzy, koniec polecenia bez wypisywania odpowiedzi

Symbole operatorów

%	Początek linii komentarza
%	{ tu się zaczyna długi komentarz….. …a tu się kończy %}
Ctrl-r	wzięcie w komentarz zaznaczonego tekstu
Ctrl-t	sciągnięcie komentarza
:	Generowanie wektorów, indeksowanie macierzy
'	Początek i koniec wprowadzania łańcuchów znakowych, transpozycja macierzy, sprzężenie macierzy
!	Komenda sytemu operacyjnego

Zmienne specjalne i stałe

ans Zmienna robocza, automatycznie przyjmuje daną wartość, jeśli nie nadano jej nazwy

computer Nazwa komputera, na którym działa Matlab

eps Precyzja zmiennoprzecinkowa

i, j Jednostka liczby urojonej

inf Nieskonczoność

NaN Wartość nieokreślona (zwykle oznacza wprowadzenie wartości nieliczbowej jako argumentu funkcji matematycznej)

nargin Liczba argumentów wejściowych funkcji

nargout Liczba argumentów wyjściowych funkcji

pi 3.1415926....

realmax Największa dostępna liczba rzeczywista

realmin Najmniejsza dostępna liczba rzeczywista

Podstawowe funkcje matematyczne

abs	Wartość bezwzględna, moduł liczby zespolonej, wektor wartości znaków łańcucha
acos, acosh	Arcus cosinus, arcus cosinus hiperboliczny
acot, acoth	Arcus cotangens, ...
acsc, acsch	Arcus cosecans, ...
angle	Kąt fazowy dla liczby zaspolonej w radianach
asec, asech	Arcus secans, ...
asin, asinh	Arcus sinus, ...
atan, atanh	Arcus tangens, ...
atan2	Arcus tangens, wynik w przedziale [-p, p]
ceil	Zaokrąglenie w góre, sufit
conj	Liczba sprzężona do liczby
cos, cosh	Cosinus, ...
cot, coth	Cotangens, ...
csc, csch	Cosecans, ...

Podstawowe funkcje matematyczne

exp e do potęgi argumentu
fix Zaokrąglenie w kierunku zera
floor Zaokrąglenie w dół, **round** Zaokrąglenie do najblizszej liczby całkowitej
gcd Największy wspólny podzielnik
lcm Najmniejsza wspólna wielokrotność
log Logarytm naturalny argumentu; **log10** Logarytm dziesiętny argumentu
real Część rzeczywista liczby zespolonej, **imag** Część urojona liczby zespolonej
rem Reszta z dzielenia
sec, sech Secans, ...
sign Znak funkcji
sin, sinh Sinus, ...
sqrt Pierwiastek kwadratowy
tan, tanh Tangens, ...

Polecenia, tworzenie macierzy

Większość **poleceń** Matlaba ma postać:
[A B C …] = polecenie(a, b, c,,…)
Gdzie A, B, C oznaczają zmienne wyjścia (wyniki polecenia), a, b, c argumenty wejściowe. Nazwy zmiennych powinny się zaczynać od liter, nie mogą zawierać spaxji, z polskich znaków, znaków zastrzeżonych. Matlab **rozróżnia wielkość liter**.

Podstawową formą zmiennych w Matlabie jest macierz (stad nazwa Matlab: Matrix Laboratory). Szczególnym przypadkiem macierzy są wektory, czyli macierze jednokolumnowe lub jednowierszowe.

Macierze (wektory) można tworzyć przez:
- wpisywanie elementów z linii komend
- generowanie elementów
- budowanie z innych macierzy
- mieszanie poprzednich sposobów

Tworzenie wektorów

Wpisywanie elementów:

Wektor poziomy N = [2 4 8]
Wektor pionowy M = [2; 4; 8]

Generowanie elementów:
-podajemy pierwszy i ostatni element oraz wartość skoku:

X = [1:0.5:4]

-funkcja linspace:

Y = linspace(3,14,17) oraz

funkcja logspace:

Z = logspace(1,2,8).

(wektor Z będzie zawierać 8 elementów od 10^1 do 10^2 z odstępami logarytmicznymi).

Operacje na macierzach możemy podzielić na dwie grupy:

•operacje macierzowe – wykonywane na całych macierzach zgodnie z regułami algebry

•operacje tablicowe – wykonywane na poszczególnych elementach macierzy

operacja	macierzowa	tablicowa	uwagi
dodawanie	+	+	
odejmowanie	–	–	
mnożenie	*	.*	
potęgowanie	^	.^	
dzielenie prawostronne	/	./	A./B => A(i,j)/B(i,j)
dzielenie lewostronne	\	.\	A.\B => B(i,j)/A(i,j)

Przykłady operacji

$$A=\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad B=\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

Uwagi:

A + B; A-B % obydwie macierze muszą posiadać te same wymiary, Można dodać/odjąć wielkość skalarną do dowolnej macierzy

A * B Liczba kolumn macierzy A musi być równa liczbie wierszy macierzy B.

5*A Mnożenie macierzy przez skalar (każdy element *5)

A .* B - każdy element macierzy A zostaje przemnożony przez odpowiadający mu element macierzy **B**. Macierze muszą mieć ten sam wymiar

A / B =**A***$\mathbf{B}^{-1}$ ($\mathbf{B}^{-1}$ = inv(**B**)), aby wykonać taką operację macierz B musi być macierzą kwadratową)

A ./ B ; **A .\ B** A i B muszą mieć te same wymiary.

Inne funkcje i operatory

`– transpozycja macierzy (zamiana wierszy na kolumny)

w przypadku macierzy o wartościach zespolonych:

` – transpozycja ze sprzężeniem

.` – sama transpozycja

det(A) - wyznacznik macierzy

inv(A) - macierz odwrotna do macierzy A

size(A) - wyświetla rozmiar macierzy A (liczb wierszy i kolumn)

ndims(A) – wyświetla liczbę wymiarów macierzy A

max(A) - zwraca największy element wektora

min(A) -zwraca najmniejszy element wektora A

sum(A -zwraca sum elementów wektora A

prod(A) - zwraca iloczyn elementów wektora A

mean(A) - zwraca średnią elementów wektora A

rot90(A) – obraca macierz A o 90^{o} przeciwnie do ruchu wskazówek zegara

diag(A) – wyznacznik macierzy

Uwaga: Operacje **max, min, sum, prod, mean** - wykonane na macierzach zwracają wektory wierszowe zawierające elementy będące wynikami tych operacji dla każdej kolumny, np. maksymalna wartość w każdej kolumnie.

'Wyciąganie' fragmentów macierzy

Jeśli mamy macierz D, która składa się z N wierszy i M kolumn, możemy wyciągać z niej interesujące nas fragmenty:

- pojedynczy wyraz d35 = A(3,5) %z trzeciego wiersza w piątej kolumnie
- wiersz N = D(4,1:M) lub N=D(4,:) %cały czwarty wiersz
- kolumnę M = D(1:N,7) lub N=D(:,7) %cała siódma kolumna
- fragment macierzy D = D(1:4,3:6) %wiersze od 1 do 4 oraz kolumny od 3 do 6
-cała macierz D lub D(:,:)

Generowanie macierzy

zeros(n,m) - wygenerujemy macierz, o n wierszach i m kolumnach, składającą się z samych zer.
ones(n,m) - wygenerujemy macierz n x m składającą się z jedynek
eye(m, n) - generujemy macierz z jedynkami na głównej przekątnej
rand(n,m) - wygenerujemy macierz nxm o wyrazach podlegających rozkładowi jednostajnemu z przedziału (0,1)
randn(n,m) - wygenerujemy macierz o wyrazach podlegających rozkładowi normalnemu o wartości oczekiwanej 0 i odch. std. =1

Wszystkie powyższe komendy mogą być jednoargumentowe – tworzone są wówczas macierze kwadratowe.

‘Sklejanie' (konkatenacja) macierzy

cat(1,A,B) lub **[A;B]** % dokleja macierz B pod macierzą A

cat(2,A,B) lub **[A B]** % dokleja macierz B na prawo od A

Powielanie macierzy

repmat(A,n,m) – powiela macierz A n razy w pionie i m razy w poziomie

Kwadrat magiczny
A=magic(A);

Skrypty

Skrypty są zapisem sekwencji poleceń wydawanych w trybie bezpośrednim, co daje możliwość powtarzania szeregu operacji. Skrypty są plikami o rozszerzeniu .m, nie mają ani parametrów wejściowych, ani wyjściowych. Operują na zmiennych w przestrzeni roboczej, mogą tworzyć tam nowe zmienne, które pozostają po zakończeniu obliczeń.

Nowy skrypt otwieramy poleceniem ***New***, pojawia się nowa grupa ikon wspomagających edycję. Dobrym nawykiem jest umowny podział na sekcje, nadawanie im tytułów, a zwłaszcza komentowanie poszczególnych linii skryptu.

Skrypty

Prawym klawiszem myszki można otworzyć menu, z którego wybieramy ***Insert Tekst Markup***, a następnie ***Document Title and Introduction***. Pojawia się pole komentarza, w którym możemy wpisać tytuł i dodatkowy komentarz. Znak **%%** oznacza podział na sekcje (podświetlane na żółto). Pojedynczy **%** na początku linii (Ctrl-r, odwołanie Ctrl-t) powoduje, że polecenie w niej zawarte nie zostanie wykonane.

```
1    %% Program sinusik (to jest nagłówek)
2    % sinusik tablicuje funkcję sinus i zapisuje w pliku
3    % Autor: ZG
4    %Data: 2021
5    %% To jest pierwsza sekcja
6 -  clear all; close all; clc;
```

Nowy skrypt przy pierwszym uruchomieniu (ikonka ***Run***) trzeba nazwać. Przy kolejnych uruchomieniach Matlab automatycznie zapisuje aktualny skrypt, nie da się powrócić do poprzedniej wersji skryptu.

Funkcje

Funkcje - podobnie jak skrypty - są m-plikami, ale w odróżnieniu od nich przyjmują argumenty wejściowe i zwracają argumenty wyjściowe. Zmienne funkcji mają charakter lokalny.

Plik funkcji różni się od pliku skryptu pierwszą linią, w której musi się znaleźć definicja funkcji w postaci:

function [x, y]=nazwa(a, b, c)

gdzie: function - słowo kluczowe (z małej litery),

x, y – argumenty wyjściowe,

nazwa – nazwa funkcji (musi być taka sama jak nazwa pliku),

a, b, c – argumenty wyjściowe

Przykład: MojaFunkcja.m

Jeśli zmienna wyjściowa jest tylko jedna, to zapis można uprościć:

function x = nazwa(a, b, c)

A jeśli nie ma zmiennych wyjściowych, to zapis funkcji wygląda następująco:

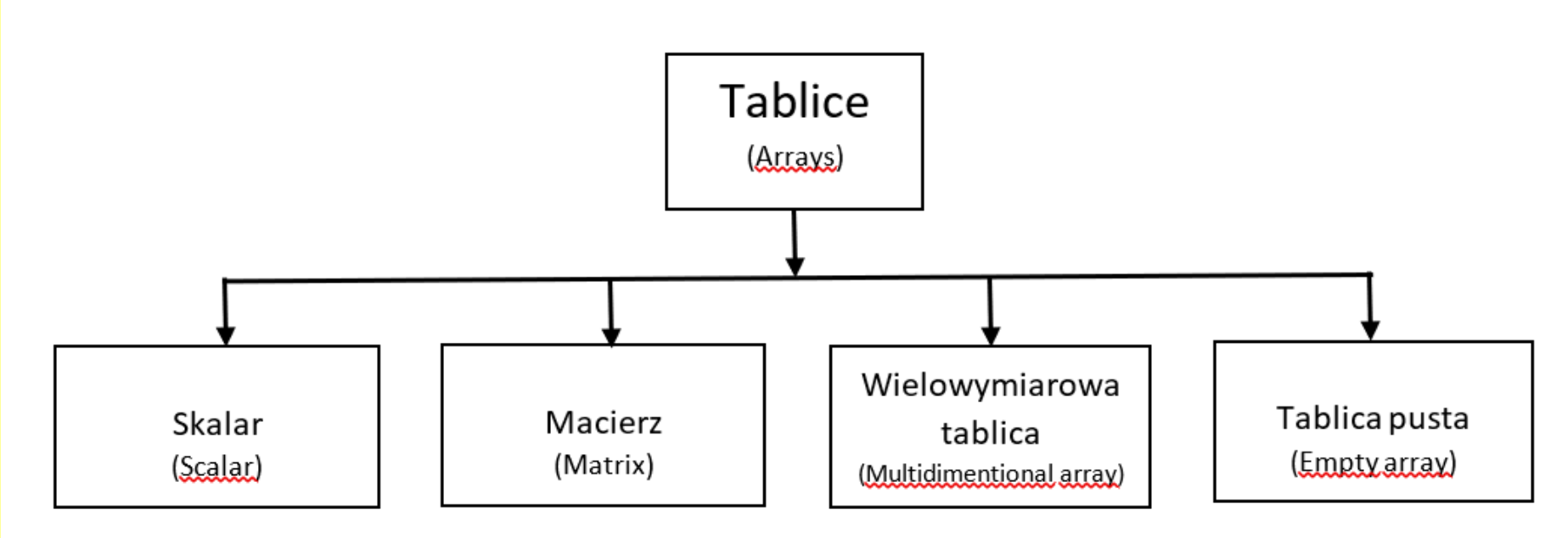
function [] = nazwa(a, b, c) lub nawet **function nazwa(a, b, c)**

Formaty wyświetlania danych

Polecenie	Wartość	Opis
Format short	3.1416	5 cyfr, reprezentacja stałoprzecinkowa
Format long	3.14159265358979	15 cyfr, reprezentacja stałoprzecinkowa
Format shortE	3.1416e+000	5 cyfr, reprezentacja zmiennoprzecinkowa
Format longE	3.141592653589793e+000	15 cyfr, reprezentacja zmiennoprzecinkowa
Format shortG	3.1416	5 cyfr, reprezentacja stało- lub zmiennoprzecinkowa
Format longG	3.14159265358979	15 cyfr, reprezentacja stało- lub zmiennoprzecinkowa
Format hex	400921fb54442d18	Liczba w układzie szesnastkowym
Format bank	3.14	2 liczby dziesiętne, np. złoty i grosze
Format rat	355/113	Przybliżona wartość liczby w postaci ułamka
Format +	+	Informacja o znaku liczby

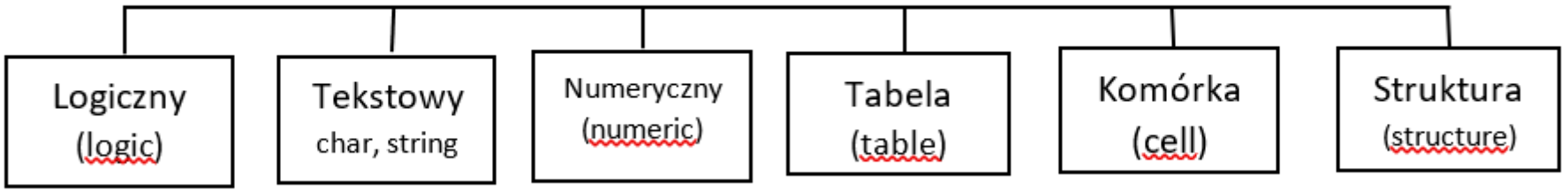
Struktura danych

Podstawową, predefiniowaną strukturą danych w Matlabie jest tablica, która może przyjmować postać skalara (tablica 1x1), macierzy, tablicy wielowymiarowej i tablicy pustej.



Typy (klasy) danych

Każdy typ tablicy może zawierać określony typ danych (logiczne, numeryczne, tekstowe, tabele, komórki, struktury i inne).

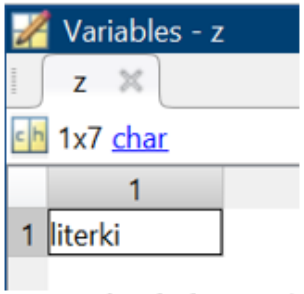


Podstawowy typ numeryczny to **double** (zmiennoprzecinkowy podwójnej precyzji, 64 bity). Wystepuje też typ **single** (o pojedynczej precyzji), **uint** (liczby całkowite bez znaku), **int** (liczby całkowite ze znakiem).

Tryb tekstowy występuje w dwóch postaciach: **char** (tablice znaków pojedynczych) i string (tablice ciągów).

Tworzymy zmienną znakową np.z używając symbolu ‘ ‘ .

z=’literki’

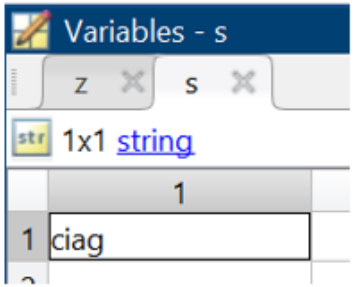


Dostęp do dowolnego znaku poprzez indeks: z(5)=’r’. Powstała nowa zmienna ans =’r’, również typu char.

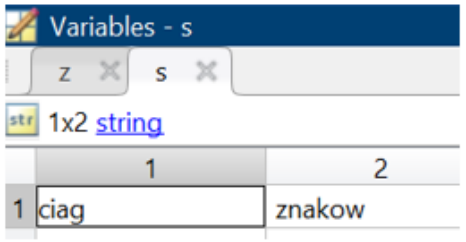
Można dodać znaki: z=[z ' male']

z = 'literki male'

Tworzymy string s=”ciag” (uwaga cudzysłów!)

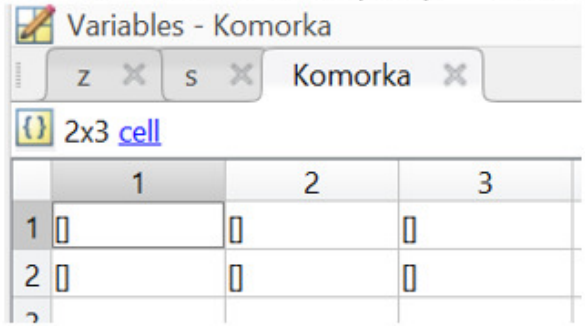


Tym razem s(1) to cały „ciag”. Można dodać kolejny fragment: s(2)=” znakow” i teraz mamy natępujące s:

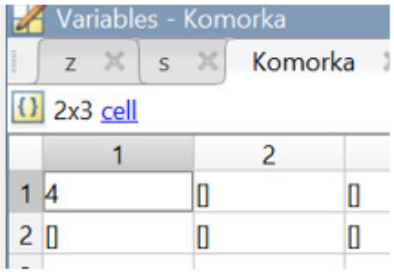


Komórki (*cell arrays*) umożliwiają przechowywanie innych typów danych o różnych rozmiarach. Dostęp odbywa się poprzez indeksowanie. Tworzymy cell array funkcją *cell*.

>> Komorka=cell(2,3) Powstała niewypełniona komórka 2x3:



>> Komorka{1,1}=4 (uwaga: nawias klamrowy)----→



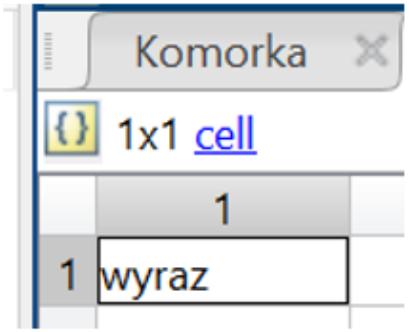
Łatwiej jednak od razu wpisać elementy np. liczby, wektor, wartość logiczną, tekst:'

>> Komorka={2,3,4;[2 3 4], false,'wyraz'}

2

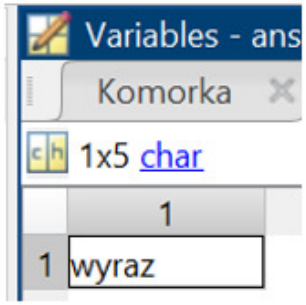
Odwoływanie się do zawartości (2 sposoby):

1. >> Komorka(2,3)



Nowo utworzona komórka też jest komórką (ans 1x1 cell)

2. Jeśli chcemy utworzyć zmienną o takim typie jak zawartość komórki, to:



a to >> Komorka{2,2:3} daje dwie zmienne po kolei:

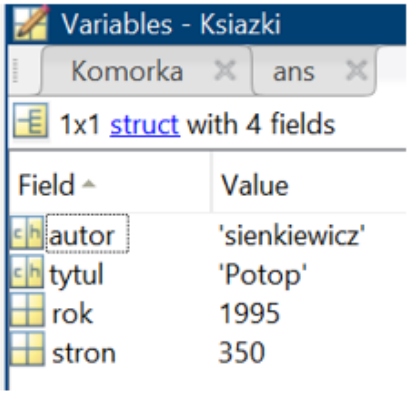
```
ans =

  logical

    0


ans =

    'wyraz'
```

Struktury.

Umożliwiają przechowywanie różnych typów danych w sposób bardziej uporządkowany. Dane są pogrupowane w ***polach***, a dostęp do nich odbywa się przez podanie nazwy pola, a nie przez indeksowanie.

Utwórzmy zmienną Ksiazki o kilku polach (autor, tytul, rok, strony) – nazwa pola po kropce.

```
>> Ksiazki.autor=’sienkiewicz’
>> Ksiazki.tytul='Potop'
>> Ksiazki.rok=1995
>> Ksiazki.stron=350
>> Ksiazki.stron(2)=400   %tak można dopisać kolejny element
```



Odwoływanie się: >> Ksiazki.tytul

ans = 'Potop'

Tabele – umożliwiają przechowywanie różnych typów danych w sposób charakterystyczny dla tabeli (jak w arkuszach): zmienne w kolumnach, każda zmienna (mogą być różnego typu) ma tyle samo wierszy. Najczęściej używane do danych statystycznych i pomiarowych.

Tabele tworzymy najczęściej przez użycie zmiennych już utworzonych lub poprzez import danych do przestrzeni roboczej Matlaba.

Przykład.

Tworzymy tabelę komendą table:

```
>> tabela1=table(LastName, Gender, Smoker, Height)   %powstała także zmienna tabela1
```

VARIABLE SELECTION

100x4 table

	1 LastName	2 Gender	3 Smoker	4 Height
1	'Smith'	'Male'	1	71
2	'Johnson'	'Male'	0	69
3	'Williams'	'Female'	0	64
4	'Jones'	'Female'	0	67
5	'Brown'	'Female'	0	64
6	'Davis'	'Female'	0	68
7	'Miller'	'Female'	1	64
8	'Wilson'	'Male'	0	68

Można też użyć funkcji readtable

>> tabela2=readtable('patients.dat')

(Plik jest tu:c:\Program Files\MATLAB\R2019b\toolbox\matlab\demos\patients.dat)

Zostały wczytane wszystkie kolumny z pliku

100x10 table

	1 LastName	2 Gender	3 Age	4 Location	5 Height	6 Weight	7 Smoker	8 Systolic	9 Diastolic	10 SelfAssessedHealthStatus
1	'Smith'	'Male'	38	'County Ge...	71	176	1	124	93	'Excellent'
2	'Johnson'	'Male'	43	'VA Hospital'	69	163	0	109	77	'Fair'
3	'Williams'	'Female'	38	'St. Mary''s ...	64	131	0	125	83	'Good'
4	'Jones'	'Female'	40	'VA Hospital'	67	133	0	117	75	'Fair'

Można importować przy pomocy kreatora importu. Znajdujemy patients.dat, klikamy prawym klawiszem myszki, wybieramy import data. Otwiera się kreator importu. Można wybrać separator (przecinek), typ zmiennej pliku wynikowego i zaznaczyć kawałek (lub wszystko). Klikamy Import Data.

VARIABLE SELECTION EDIT

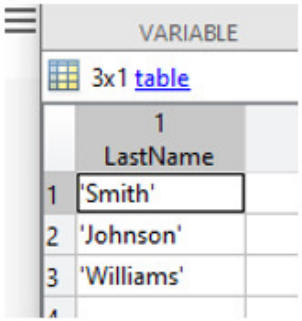
37x6 table

	1 LastName	2 Gender	3 Age	4 Location	5 Height	6 Weight
1	"LastName"	Gender	NaN	Location	NaN	NaN
2	"Smith"	Male	38	County Gen...	71	176
3	"Johnson"	Male	43	VA Hospital	69	163
4	"Williams"	Female	38	St. Mary's ...	64	131
5	"Jones"	Female	40	VA Hospital	67	133
6	"Brown"	Female	49	County Gen...	64	119
7	"Davis"	Female	46	St. Mary's ...	68	142
8	"Miller"	Female	33	VA Hospital	64	142
9	"Wilson"	Male	40	VA Hospital	68	180
10	"Moore"	Male	28	St. Mary's ...	68	183
11	"Taylor"	Female	31	County Gen...	66	132

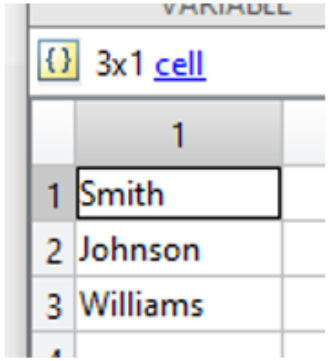
Odwoływanie się do elementów tabel.

1. Przez indeksy i nawias okrągły () - powstaje nowa tabela (tabela zawierająca pierwsze 3 elementy pierwszej kolumny)

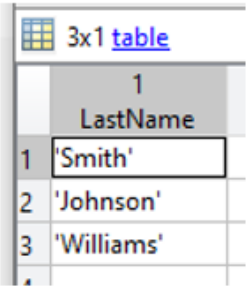
```
>> tabela1(1:3,1)
```



2. Przez indeksy i nawias {} – powstaje zmienna o takim samym typie, w tym przypadku cell

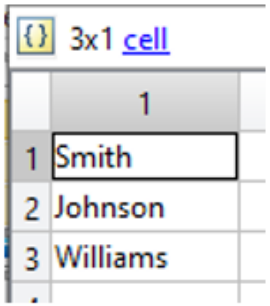


3. Przez nazwy zmiennej np. tabela1(1:3,'LastName') – efekt jest taki sam



Można kilka zmiennych: tabela1(1:3,{'LastName','Gender'}) – uwaga na {}

4. Z operatorem .
   >> tabela1.LastName(1:3) - taki sam efekt jak w metodzie 2



Tabele umożliwiają nadawanie nazw wierszom.

```
>> tabela1=table(LastName, Gender, Smoker, Height,'RowNames',LastName)
```

100x4 table

	1 LastName	2 Gender	3 Smoker	4 Height	5
1 Smith	'Smith'	'Male'	1	71	
2 Johnson	'Johnson'	'Male'	0	69	
3 Williams	'Williams'	'Female'	0	64	
4 Jones	'Jones'	'Female'	0	67	
5 Brown	'Brown'	'Female'	0	64	

Teraz i kolumny i wiersze mają swoje nazwy.

Można się odwołać po nazwie wiersza (lub kilku):

```
>> tabela1({'Brown','Smith'},:)
```

2x4 table

	1 LastName	2 Gender	3 Smoker	4 Height
1 Brown	'Brown'	'Female'	0	64
2 Smith	'Smith'	'Male'	1	71

Po wierszach i kolumnie, np. wybrany parametr dla wybranych pacjentów:

```
>> tabela1{{'Brown','Smith'},'Gender'}
```

2x1 cell

	1	2
1	Female	
2	Male	

Zapisywanie i odczytywanie danych z plików

Podstawowe funkcje zarządzania katalogami:

pwd – bieżący katalog

dir, ls – lista plików w bieżącym katalogu, dozwolone są maski*

cd<katalog>, cd.. – wejście do podkatalogu, wyjście o jeden poziom do góry

delete <plik> – kasowanie pliku

! polecenie – polecenie systemu, np. !rd<katalog> usuwa katalog

DZIĘKUJĘ ZA UWAGĘ